

An algorithm for Boolean satisfiability based on generalized orthonormal expansion

Virendra Sule

Department of Electrical Engineering
Indian Institute of Technology Bombay, India
vrs@ee.iitb.ac.in

July 16, 2014

Abstract

This paper proposes an algorithm for deciding consistency of systems of Boolean equations in several variables with co-efficients in the two element Boolean algebra $B_0 = \{0, 1\}$ and find all satisfying assignments. The algorithm is based on the application of a well known generalized Boole-Shannon orthonormal (ON) expansion of Boolean functions. A necessary and sufficient consistency condition for a special class of functions was developed in [11] using such an expansion. Paper [11] develops a condition for consistency of the equation $f(X) = 0$ for the special classes of Boolean functions 1) f in $B(\Phi(X))$ for an ON set Φ of Boolean functions in X over a general Boolean algebra B and 2) f in $B(X_2)(\Phi(X_1))$. The present paper addresses the problem of obtaining the consistency conditions for arbitrary Boolean functions in $B_0(X)$. Next, the consistency for a single equation is shown equivalent to another system of Boolean equations which involves the ON functions and characterizes all solutions. This result is then extended for Boolean systems in several variables over the algebra $B_0 = \{0, 1\}$ which does not convert the system into a single equation. This condition leads to the algorithm for computing all solutions of the Boolean system without using analogous resolution and determine satisfiability. For special systems defined by CNF formulas this algorithm results into an extension of the DPLL algorithm in which the *splitting rule* is generalized to several variables in terms of ON terms in the sense that splitting of CNF set in a single variable x is equivalent to ON terms x, x' .

Category: cs.CC, cs.SC, ms.RA

ACM class: I.1.2, F.2.2, G.2

MSC class: 03G05, 06E30, 94C10.

1 Introduction

The problem of Boolean satisfiability over the Boolean algebra $B_0 = \{0, 1\}$ is defined by a system of equations

$$f_i(X) = g_i(X), i = 1, 2, \dots, N \quad (1)$$

where $f_i(X), g_i(X)$ are Boolean functions of n Boolean variables x_1, \dots, x_n denoted X with B_0 co-efficients where $B_0 = \{0, 1\}$ is the well known two element Boolean algebra. Boolean functions in n variables X with co-efficients in a general Boolean algebra B are formal expressions

with Boolean operations $+, \cdot, '$ between variables, expressions and constants in B . Equivalence classes of these expressions, having same B values when variables are assigned from B are called Boolean functions. These form a Boolean algebra which we shall denote alternatively by $B(X)$ and also by $B(n)$. For definition and properties of different types of Boolean functions we refer [3, 2]. The system (1) is said to be *consistent* or *satisfiable* if there exist values (called *assignments*) of variables X in B_0^n at which the equations hold true.

In order to introduce the problem addressed in this paper we need to consider previous results and background. A set $\Phi = \{\phi_1, \dots, \phi_m\}$ in a Boolean algebra B is said to be *orthogonal* (OG) of order m if $\phi_i \phi_j = 0$ for $i \neq j$ and is called *orthonormal* (ON) if in addition to this condition the members satisfy

$$\sum_{i=1}^m \phi_i = 1$$

Such a set is said to be *reduced* if none of the members ϕ_i are zero in B . In this paper by an ON set we shall always refer to a reduced ON set. For a background of ON systems in general Boolean algebra the reader is referred to [3]. If Φ is an ON set of order m in the algebra of Boolean functions $B(X)$, by the well known result [2, proposition 3.14.1] a Boolean function $f(X)$ in $B(X)$ has an expansion as

$$f(X) = \sum_{\phi_i \in \Phi} \alpha_i(X) \phi_i(X) \quad (2)$$

This expansion generalizes the well known Shannon expansion (proposed to be called Boole-Shannon expansion in [11]). The expansion co-efficient functions $\alpha_i(X)$ for a given ON set Φ are not unique as for every i any function in the range $[f\phi_i, f + \phi_i']$ satisfies the expression.

1.1 Previous results

Given an ON set Φ in $B(X)$ the set $B(\Phi)$ is defined in [11] by

$$B(\Phi) = \left\{ \sum_{\phi_i \in \Phi} \alpha_i \phi_i, \alpha_i \in B \right\}$$

The problem addressed in [11] is to express the condition for consistency of a single Boolean equation $f(X) = 0$ for f in $B(\Phi)$ in terms of the expansion co-efficients α_i . It is shown in [11, Theorem 2] that $f(X) = 0$ for such a function is consistent iff

$$\prod_i \alpha_i = 0$$

in B . Implications of this result in terms of elimination of variables is discussed in [11] and leads to a procedure for deciding consistency by eliminating partial set of variables successively. If the function f belongs to $B(X)$ in n variables X while Φ are defined over a subset X_1 in a partition $X = \{X_1, X_2\}$ then if f belongs to $B(X_2)(\Phi(X_1))$ the expansion co-efficients α_i exist in $B(X_2)$. Under these conditions it is further shown in [11, Corollary 1] that $f = 0$ is consistent iff

$$\prod_i \alpha_i(X_2) = 0 \quad (3)$$

is consistent. This way the number of variables can be eliminated successively to finally arrive at a condition for consistency of $f = 0$. The product of co-efficients above is analogous to the well known resolution of clauses in CNF SAT studies [5].

1.2 Problems addressed

The aim of this paper is twofold. First we want to determine the consistency conditions for $f(X) = 0$ w.r.t. an ON set Φ in $B(X)$ for general functions (not restricted to $B(\Phi)$). Due to this generality an ON expansion of a function $f(X)$ w.r.t. Φ may have the co-efficients $\alpha_i(X)$ as functions which may not be constants in B . Moreover as a functions in $B_0(X)$ the arguments of co-efficients and ON functions might also overlap. Previous proof of consistency [11] considers the case when only the constant co-efficients α are involved or the when the arguments of all co-efficients α_i in the expansion are distinct from those of the ON set Φ . Hence this proof is not applicable in the present case.

Further, the aim of this paper is to develop an algorithm for deciding Boolean satisfiability (and to determine all solutions) of Boolean *systems* (1) where functions in the system belong to $B_0(X)$. Determining all solutions of a Boolean system is interpreted in the sense that the original Boolean system is decomposed in terms of multiple systems in smaller number of variables whose union is a projection of solution set of the original system. Hence the original system is consistent iff at least one of the smaller systems is consistent. This decomposition can be iteratively applied to finally get in principle smallest Boolean systems involving no variables hence of the form $a = b$ for a, b in B_0 which decide consistency. While iteratively decomposing the original Boolean system, several trivial assignments of variables can be discovered (such as those involving unit clauses $x = 1$ or $x' = 1$ as well as pure literals as in CNF satisfiability by DPLL algorithm). Such a decomposition strategy for Boolean systems is desirable for efficient computation of satisfiability and solutions.

1.2.1 Computational aspects

To explain the computational aspects we first note that algorithms for Boolean satisfiability by elimination involve two impediments

1. The consistency condition or the process of elimination of variables involves computation of eliminant of a single equation $f(X) = 0$. Hence it is necessary to convert a system of Boolean equations (1) into such an equation by constructing the single associated function

$$F(X) = \sum_{i=1}^N (f_i(X) \oplus g_i(X))$$

Hence an algorithm for consistency of the system has to handle an overload of computing $F(X)$ which may be heavy in memory requirement.

2. The resolution step involves computation of the product (3) whose consistency needs to be determined. Computation of this product is another (memory) overhead of such algorithms.

Due to these difficulties elimination of variables for computation may be advisable only when number of variables to be eliminated is small enough relative to memory available. Hence it is desirable for an algorithm for satisfiability of a system (1) to avoid computation of the single equivalent equation $F = 0$ as well as computation of the resolution product. We show in this paper that the ON expansion based algorithm can be constructed for satisfiability with these advantageous features for systems of Boolean equations whose functions belong to $B_0(X)$. Apart from these benefits we show that the algorithm has advantages such as

1. The algorithm is applicable for decomposition of systems without any special representation of the problem such as in CNF or DNF. For CNF satisfiability the algorithm can provide a decomposition which generalizes the splitting rule of the well known DPLL algorithm.
2. The algorithm in principle computes all solutions of the system whenever they exist otherwise it returns un-satisfiability.
3. The algorithm provides a natural decomposition of a bigger system into smaller systems to be solved independently. Hence this algorithm has an inherent parallelism.

1.3 SAT literature, elimination and relations with known approaches

Well known problems of CNF or DNF satisfiability, commonly referred as SAT problems are special cases of the Boolean satisfiability problem. SAT problems have been investigated in Computer Science for several decades due to their vast applications to Propositional Logic, Artificial Intelligence, software and hardware verification, Operational Research and in recent times to cryptology. The vast bibliographies of [5, 6] and topics covered show the breadth and depth of research carried out in theory and algorithms for Boolean satisfiability in particular for systems represented in CNF and DNF. Much of the progress in SAT theory and practice is based on the celebrated DPLL algorithm and more modern approaches such as GRASP [5] as well as complete and incomplete algorithms. Specialized methods for SAT are a focus of developments reported in [6, 7, 8]. In cryptology and computer algebra, algorithms based on extension of Gaussian elimination called XL method and Grobner basis computation have attracted considerable attention [7] where Boolean systems are often represented in the polynomials in several variables with B_0 co-efficients considered as the binary field $GF(2)$. Consistency of Boolean systems of equations discussed in [3, 2] on the other hand explain the method of elimination of variables. Elimination of variables in CNF SAT problems is equivalent to resolution of clauses. Due to high memory requirements early algorithms for CNF SAT which used resolution were modified to avoid resolution leading to the modern version known as DPLL. GRASP on the other hand is a different approach than DPLL which explores assignment substitutions along with incorporation of conflict resolving clauses.

ON expansion based algorithm developed in this paper is essentially equivalent to assignment substitutions decided by an ON set of functions. In certain SAT algorithms assignments of unknown variables are used to reduce the problem size, but can create conflicts which are resolved by going back and correcting the assignments. In ON expansion based assignments no conflicts arise. These are also equivalent to a several variable analogue of the splitting rule of DPLL at a single variable. One of the areas of current research in SAT algorithms and applications is development of parallel algorithms which are scalable for large data problems

and for large number of processors. Recent survey [9] shows the state of the art and challenges in development of parallel methods for SAT. An important direction in parallel algorithm development is decomposition of the Boolean system (or data set of CNFs or DNFs in SAT problems) for parallel processing. An advantages of ON based algorithm, as will be clear from this paper, is the natural decomposition of the problem it provides due to the ON expansion. Another issue with SAT algorithms is that they essentially compute only one solution if such exists or return un-SAT result. ON based algorithm on the other hand can compute all solutions when satisfiability holds, characterized by supports of the ON functions used. As an illustration we show computation of all rational solutions of an elliptic curve in a finite field.

2 Zero sets of Boolean functions and correspondence with algebra

Before discussing the consistency conditions we shall briefly present background results which shall be useful later at several places. The set $B_0(X)$ of functions in n variables X is a Boolean algebra w.r.t. Boolean operations in functions: For f, g in $B_0(X)$ Boolean operations $\{+, \cdot, '\}$ are defined with the help of values of functions in the Boolean algebra B_0 (whose Boolean operations are denoted by same symbols) on points x in B_0^n as follows:

$$\begin{aligned}(f + g)(x) &= f(x) + g(x) \\ (fg)(x) &= f(x)g(x) \\ f'(x) &= f(x)' \\ (f \oplus g)(x) &= f(x) \oplus g(x)\end{aligned}$$

For the Boolean inequality $f \leq g$ in $B_0(X)$ there is the following equivalence.

$$\Leftrightarrow \begin{array}{l} f \leq g \\ f(x)g(x)' = 0 \quad \forall x \in B_0^n \end{array}$$

We now relate the Boolean functions with their zeros and supports which are formally denoted as follows:

1. The *zero set* of a function f in $B_0(X)$, denoted $V(f)$, is the set,

$$V(f) = \{x \in B_0^n | f(x) = 0\}$$

2. The *support* of a function f , denoted $\text{supp}f$ is the set

$$\text{supp}f = \{x \in B_0^n | f(x) = 1\}$$

$\text{supp}f$ is also often called the *set of one values* in the literature [6].

For f in $B_0(X)$ we have the obvious identity with respect to set complement in B_0^n

$$V(f)^c = \text{supp}f \tag{4}$$

The relationship of zeros with the algebraic properties is given by the following

Proposition 1. For f, g in $B_0(X)$

$$\begin{aligned}
1) \quad V(f+g) &= V(f) \cap V(g) \\
2) \quad V(f') &= V(f)^c = \text{supp} f \\
3) \quad V(fg) &= V(f) \cup V(g) \\
4) \quad f \leq g &\text{ iff } V(g) \subset V(f) \\
&\Leftrightarrow \text{supp} f \subset \text{supp} g
\end{aligned} \tag{5}$$

Proof: 1) $V(f+g) = \{x \in B_0^n | f(x) + g(x) = 0\} = \{x \in B_0^n | f(x) = g(x) = 0\} = V(f) \cap V(g)$.
2) $V(f') = \{x \in B_0^n | f(x)' = 0\} = \{x \in B_0^n | f(x) = 1\} = \text{supp} f = V(f)^c$.
3) $V(fg) = V((f' + g')') = V(f' + g')^c = (V(f') \cap V(g'))^c = V(f')^c \cup V(g')^c = V(f) \cup V(g)$.
4) This is established in following equivalences

$$\begin{aligned}
f \leq g &\Leftrightarrow fg' = 0 \text{ in } B_0(X) \\
&\Leftrightarrow f(x)g(x)' = 0 \forall x \in B_0^n \\
&\Leftrightarrow f(x) \leq g(x) \forall x \in B_0^n \\
&\Leftrightarrow \text{supp} f \subset \text{supp} g \\
&\Leftrightarrow V(g) = (\text{supp} g)^c \subset (\text{supp} f)^c = V(f)
\end{aligned}$$

□

A point $a = (a_1, \dots, a_n)$ in B_0^n is the zero set $V(f_a)$ of the function

$$\begin{aligned}
f_a(x_1, \dots, x_n) &= \sum_{i=1}^n (x_i \oplus a_i) \\
&= \sum_{i=1}^n (x_i a'_i + x'_i a_i) \\
&= \prod_{i=1}^n (x_i + a_i)(x'_i + a'_i)
\end{aligned} \tag{6}$$

For points in B_0^n we have another involutory operation *star* defined as follows: Let $a = (a_1, \dots, a_n)$ then $a^* = (a'_1, \dots, a'_n)$. Let $b = (b_1, \dots, b_n)$ be another point in B_0^n . The *dual* of a function $f(X)$ denoted $f^d(X)$ is defined as

$$f^d(X) = f(X^*)' \tag{7}$$

We have following relationships which can be proved easily.

$$\begin{aligned}
1) \quad a \cup b &= V(f_a f_b) \\
2) \quad a^* &= V(((f_a)^d)') = \text{supp}(f_a)^d \\
3) \quad V(f)^* &= \text{supp} f^d
\end{aligned} \tag{8}$$

Finally, for convenience we recall some of the well known properties of co-efficients in ON expansions. For Boolean functions f, g in $B(X)$, an ON set of functions Φ and ON expansions (2) for f and

$$g(X) = \sum_{\phi_i \in \Phi} \beta_i(X) \phi_i(X)$$

following identities hold [2]

$$f(X) + g(X) = \sum_{\phi_i \in \Phi} (\alpha_i(X) + \beta_i(X))\phi_i(X) \quad (9)$$

$$f(X)g(X) = \sum_{\phi_i \in \Phi} (\alpha_i(X)\beta_i(X))\phi_i(X) \quad (10)$$

$$f(X)' = \sum_{\phi_i \in \Phi} \alpha_i(X)'\phi_i(X) \quad (11)$$

$$f(X) \oplus g(X) = \sum_{\phi_i \in \Phi} (\alpha_i(X) \oplus \beta_i(X))\phi_i(X) \quad (12)$$

Composition of Boolean functions can also be expressed in ON expansion as follows. The simple proof is omitted. Let $g_i(X)$ for $i = 1, \dots, n$, $f(X)$ be Boolean functions in n -variables and

$$g_i(X) = \sum_j \beta_{ij}(X)\phi_j(X)$$

be expansions of $g_i(X)$ in the ON set as above. Then

$$f(g_1(X), \dots, g_n(X)) = \sum_j \alpha_j(\beta_{1j}(X), \dots, \beta_{nj}(X))\phi_j(X)$$

This identity can be arrived at from previous identities for elementary Boolean operations between functions.

3 Extension of the consistency condition

We now take up the problem of extending the result of consistency of $f(X) = 0$ obtained in [11] to the general case when the function f need not be in $B(\Phi)$ for an ON set in $B(X)$. But we take up the special case in which B is itself of the form $B_0(X_2)$ hence the satisfiability is in terms of $\{0, 1\}$ assignments. Although mathematically this is still a greatly restrictive formulation, it is nevertheless of considerable interest to Computer Science. In order to highlight the problem of consistency, consider a function $f(X)$ and Φ an ON set of functions of order m both in $B_0(X)$. Consider an ON expansion (2) of $f(X)$. The questions that we want to investigate in this section are,

"What conditions on the co-efficient functions $\alpha_i(X)$ arising in the expansion are necessary and/or sufficient for consistency of $f(X) = 0$? How do such conditions change with the ON set? Further, since the expansion co-efficient functions $\alpha_i(X)$ are not unique whether such conditions can be more advantageous as compared to any selection for a special choice of these co-efficients?"

Note that these questions were answered for the special case when f belonged to $B(\Phi)$ for a general Boolean algebra B in [11, Theorem 2, Corollary 1, Corollary 4] but in the present case we are considering general functions $f(X)$ in $B_0(X)$ hence the functions $\phi_i(X)$ in the ON set and the co-efficient functions $\alpha_i(X)$ may share common variables. We begin first with a general case for which we have an obvious result,

Proposition 2. Following two conditions hold in respect of an ON expansion (2)

1. (Necessity). If $f(X) = 0$ is consistent then there exists an index i such that

$$\alpha_i(X) = 0 \quad (13)$$

is consistent.

2. (Sufficiency). If the equation in co-efficients

$$\sum_{i=1}^m \alpha_i(X) = 0 \quad (14)$$

is consistent then $f(X) = 0$ is consistent.

Proof: Sufficiency is obvious from the ON expansion for if there exists x in B_0^n such that $\alpha_i(x) = 0$ for all i then $f(x) = 0$.

We thus prove the necessary condition. Let $f = 0$ be consistent, then there exists x in B_0^n such that

$$\sum_{i=1}^m \alpha_i(x) \phi_i(x) = 0$$

As Φ is ON,

$$\sum_i \phi_i(X) = 1$$

hence there exists an index i such that $\phi_i(x) = 1$ and $\phi_j(x) = 0$ for $j \neq i$. This implies

$$0 = f(x) = \alpha_i(x)$$

and proves the necessary condition. \square

The proposition implies that consistency of at least one equation $\alpha_i(X) = 0$ for some i is necessary for consistency of $f = 0$ while consistency of the system

$$\alpha_i(X) = 0 \forall i$$

is sufficient.

Consistency of an equation $f = 0$ has important connection with the theory of *elimination* of Boolean variables. (We refer the reader to [3, 2] for a background on elimination theory). In the analysis of elimination of variables in Boolean equations, the Boole-Shannon expansion in terms of a single variable x is

$$f(X) = f_1x + f_0x'$$

(where $f_1 = f(x = 1)$, $f_0 = f(x = 0)$) which is an ON expansion relative to the ON set $\Phi = \{x, x'\}$. In this case the function f belongs to the special class $B(\{x, x'\})$ where $B = B_0(X - \{x\})$. Hence the necessary and sufficient condition of [11] is the same as the well known consistency condition in terms of the eliminant

$$f_0f_1 = 0$$

In the present general case we do not have co-efficients α_i in the expansion independent of the variables X in the function, nor do we have a necessary and sufficient condition for consistency in general. Hence the conditions of proposition 2 cannot be related to elimination of variables. But if the ON set is a set of minterms in a subset of variables in X (or all of these variables), we can get the necessary and sufficient condition as shown in the next subsection and can be related to elimination. Hence above conditions for consistency appear to be mainly of theoretical consequence and do not provide any computationally useful consequence which was the main motivation of this paper. Also the proof explicitly makes use of the fact that f belongs to $B_0(X)$. Analogous conditions can be derived for the one value form of the equation as follows.

Corollary 1. Following two conditions hold in respect of an ON expansion (2)

1. (Necessity). If $f(X) = 1$ is consistent then there exists an index i such that

$$\alpha_i(X) = 1 \quad (15)$$

is consistent.

2. (Sufficiency). If the equation in co-efficients

$$\prod_{i=1}^m \alpha_i(X) = 1 \quad (16)$$

is consistent then $f(X) = 1$ is consistent.

Proof: $f(X) = 1$ is consistent iff $f(X)' = 0$ is consistent. The ON expansion of $f(X)'$ in Φ is

$$f(X)' = \sum_{i=1}^m \alpha_i(X)' \phi_i(X)$$

Hence both conditions follow from the conditions (13), (14) respectively. \square

3.1 Special expansion: ON set of minterms

Consider now the special case when the variables X are partitioned in two subsets X_1, X_2 and the ON set $\Phi = \{\mu_i(X_1)\}$ is a set of minterms μ_i in X_1 variables. For such an ON set an expansion of f in $B_0(X)$ of the form

$$f(X) = \sum_{i=1}^m (f/\mu_i)(X_2) \mu_i(X_1)$$

can be chosen in which the co-efficients of expansion $\alpha_i = (f/\mu_i)$ are functions of the X_2 variables alone. For such a special expansion and ON functions we can prove a necessary and sufficient condition for consistency as follows.

Theorem 1. For a partition of X as described above, the special ON set Φ of minterms in X_1 and the choice of a special expansion of f as above, the equation $f(X) = 0$ is consistent iff

$$\prod_{i=1}^m \alpha_i(X_2) = 0$$

is consistent.

Proof: If $f(X) = 0$ is consistent then by the necessary condition of proposition 2 there exists an index i such that

$$\alpha_i(X_2) = 0$$

is consistent. This implies the above condition.

Conversely, if $f(X) = 0$ is not consistent, then

$$f(x) = 1 \forall x \in B_0^n$$

Let $X = (X_1, X_2)$ denotes the X_1 and X_2 variable components. For each assignment x_1 there is an i and a unique minterm $\mu_i(X_1)$ such that $\mu_i(x_1) = 1$ while $\mu_j(x_1) = 0$ for $j \neq i$. Hence

$$f(x_1, X_2) = \alpha_i(X_2)$$

which implies that

$$\alpha_i(x_2) = 1 \forall x_2$$

for all assignments x_2 . As x_1 is varied over all assignments it follows that $\alpha_i(x_2) = 1$ for all i and x_2 . Hence

$$\prod_{i=1}^m \alpha_i(x_2) = 1$$

which implies that

$$\prod_{i=1}^m \alpha_i(X_2) = 0$$

is not consistent. This proves sufficiency. \square

3.2 Computationally useful formulation of consistency

Finally, we consider a formulation of the consistency condition in terms of an ON expansion which will be useful for the purpose of computation as well as resolving the problem of consistency and computation of solutions of systems of equations.

Proposition 3. Given an ON expansion (2) of $f(X)$ the equation $f(X) = 0$ is consistent iff there exists an index i in $1, \dots, m$ such that the system

$$\begin{aligned} \alpha_i(X) &= 0 \\ \phi_i(X) &= 1 \end{aligned}$$

is consistent.

Proof: If the equation is consistent and x in B_0^n is a solution then

$$\sum_{i=1}^m \alpha_i(x) \phi_i(x) = 0$$

But as ϕ_i are ON this implies there exists i in $[1, m]$ such that $\phi_i(x) = 1$ and that $\phi_j(x) = 0$ for $j \neq i$. This implies that the system above is consistent.

To prove sufficiency, let the above condition hold for some index i and let q in B_0^n be a solution of the equations. Then, since $\phi_j(X)\phi_i(X) = 0$ for all $j \neq i$ evaluating this identity at q gives $\phi_j(q) = 0$ for all $j \neq i$. Hence $f(q) = 0$ since all terms in the (2) when evaluated at q are zero. This proves that the condition is sufficient for consistency of the equation $f(X) = 0$. \square

This formulation of consistency is important for two reasons. First, it gives a necessary and sufficient condition in the general case, second it relates consistency of the equation $f = 0$ to the consistency of a system in which one equation is in a member of the ON set. Since ON functions are well characterized [3] their solutions and one values can also be assumed well characterized, hence the system indicates a natural decomposition of the space of search for solutions. This is essentially a computational advantage and shall be developed further to tackle the problem of solving systems of Boolean equations efficiently.

3.3 Conjugate consistency problems

So far we have considered the consistency of the equation $f(X) = 0$ in terms of an ON expansion of f . There is however the associated function $f(X^*)$ in the definition of the dual function f^d whose support is the conjugate of the set of solutions $V(f)$ of this equation as illustrated in equation (8). Hence it is natural that instead of considering the problem of solving one equation $f(X) = 0$ associated problems are also explored and a symmetry is achieved in some sense. We shall call these as *conjugate problems*. The ON functions and expansion co-efficients of conjugate problems are also closely related as shown below.

Let $\Phi = \{\phi_1, \dots, \phi_m\}$ be an ON set of functions in $B_0(X)$ then the set

$$\Phi^* = \{\phi^*_1, \dots, \phi^*_m\} = \{\phi_1(X^*), \dots, \phi_m(X^*)\}$$

is also ON. To the ON expansion (2) there is an associated expansion

$$f(X^*) = \sum_{i=1}^m \alpha_i(X^*) \phi^*_i(X) \quad (17)$$

We shall call this a *conjugate expansion*. The supports of $\phi_i(X)$ and $\phi_i(X^*)$ are also conjugates i.e. for a q in B_0^n ,

$$\phi_i(q) = 1 \text{ iff } \phi^*_i(q^*) = 1$$

The Boolean equation $f(X) = 0$ has the solution set $V(f)$ in B_0^n while from 3) of (8) it follows that $V(f)^*$ is the zero set of $f(X^*)$. We shall call such problems whose solution sets are conjugates as *conjugate consistency problems*. For a single equation we have

Proposition 4. Following pairs of consistency problems are conjugate with solution sets indicated

- (1) $V(f) : f(X) = 0 \quad V(f)^* : f(X^*) = 0$
- (2) $V(f)^c : f(X) = 1 \quad V(f)^{*c} : f(X^*) = 1$

4 Elimination of variables in systems case

Let f be a function of one variable ξ in $B(\xi)$. The equation $f(\xi) = 0$ is consistent iff $f(1)f(0) = 0$ in B [2, Section 7.3.1]. This leads to the notion of elimination of variables from Boolean

equations. Elimination of a variable ξ in an equation $f(\xi, Y) = 0$ where Y are other variables in the Boolean function f in $B(\xi, Y)$ follows from the expansion

$$f(\xi, Y) = f(1, Y)\xi + f(0, Y)\xi'$$

From the consistency condition it follows that $f(\xi, Y) = 0$ is consistent iff

$$f(1, Y)f(0, Y) = 0$$

is consistent in $B(Y)$ which is a new equation involving only Y variables. The function $F(Y) = f(1, Y)f(0, Y)$ is called the *eliminant* of f w.r.t. ξ .

Now if instead of $f = 0$ there is a system of equations in variables x, Y where x is a single variable and other variables Y ,

$$f(x, Y) = 0 \tag{18}$$

$$g(Y) = 0 \tag{19}$$

then we have the consistency condition given by

Lemma 1. The system (18) is consistent iff $F(Y) + g(Y) = 0$ is consistent where $F(Y)$ is the eliminant of $f(x, Y)$ w.r.t. x .

Proof: The system (18) is consistent iff the single equation

$$f(x, Y) + g(Y) = 0$$

is consistent. By elimination of x , this equation is consistent iff

$$\begin{aligned} [f(1, Y) + g(Y)][f(0, Y) + g(Y)] &= f(1, Y)f(0, Y) + (f(1, Y) + f(0, Y))g(Y) + g(Y) \\ &= F(Y) + g(Y) = 0 \end{aligned}$$

is consistent (which is equivalent to consistency of the system $F(Y) = 0, g(Y) = 0$) where F is the eliminant of $f(x, Y)$ w.r.t. x . \square

Thus for the system (18) where the x variable is absent in g , elimination of x can be carried out to get the equation $F(Y) = 0$ independent of computation on g . However the consistency condition has another interpretation

Corollary 2. The system (18) is consistent iff $g(Y) = 0$ is consistent and there exists a solution α of $g(Y) = 0$ such that the equation $f(x, \alpha) = 0$ is consistent.

Proof: Since $F(Y) = f(1, Y)f(0, Y)$ consistency of $F(Y) + g(Y) = 0$ is equivalent to $g(Y) = 0$ being consistent and there exists α such that $g(\alpha) = 0$ and $f(1, \alpha)f(0, \alpha) = 0$ which is the consistency of $f(x, \alpha) = 0$. \square

Hence alternatively consistency of the system (18) can be obtained by evaluating f at solutions of g and determining consistency of the resultant function. This has computational implications which are discussed next.

4.1 Computation of solutions without converting to single equation form

Above corollary has an advantageous implication for computation of consistency and a solution of systems. For a system of equations above in which the variables are partitioned as in (18) in X, Y where $|X|$ is large, if the number of variables Y is small enough or the consistency and computation of solutions of $g(Y) = 0$ is easy, then the consistency of the system can be determined by decomposing the system into independent problems of consistency of $f(X, \alpha) = 0$ and $g(\alpha) = 0$. Having the variables Y assigned values α computing consistency in X variables is considerably advantageous than elimination of X with indeterminate Y . Such an approach is thus an alternative to elimination of variables X in deciding consistency and finding a solution without converting the problem to a single equation. We shall show that ON decomposition of functions in a general system of equations exploits this aspect in computation and is hence expected to provide a decomposition of systems without converting them to single equation form. Following proposition shall be a preliminary result before building up our algorithm for general systems.

We now consider the special case of functions $B_0(X)$ in n -variables. Let Φ be a set of m ON functions and consider an expansion (2) of f . For any non zero function ϕ in Φ , there exists a non-empty set Q in B_0^n such that

$$\phi(q) = 1 \forall q \in Q$$

the union of all such sets is called the set of one values or *support* of ϕ denoted $\text{supp}\phi$. Following proposition is almost a restatement of proposition 3.

Proposition 5. Let f be a Boolean function given with an ON expansion (2). Then $f(X) = 0$ is consistent iff there exists an index i and an assignment q in support of ϕ_i such that $\alpha_i(q) = 0$. Every solution of $f(X) = 0$ when consistent arises this way.

Proof: If $f(X) = 0$ is consistent there exists a solution q in B_0^n which satisfies

$$\alpha_i(q)\phi_i(q) = 0 \forall i = 1, \dots, m$$

Since $\sum \phi_i = 1$ there is an index i such that $\phi_i(q) = 1$ hence $\alpha_i(q) = 0$. This proves necessity and also shows that every solution q satisfies these conditions.

Conversely, if such an index i and q satisfying given conditions exist such that $\phi_i(q) = 1$ then $\phi_j(q) = 0$ for all $j \neq i$. Hence since $\alpha_i(q) = 0$

$$f(q) = \alpha_i(q) + \sum_{j \neq i} \alpha_j(q)\phi_j(q) = 0$$

Hence $f = 0$ is consistent and q is a solution. This shows the conditions are necessary for consistency and every such q is a solution. \square

Computational consequence of this proposition is quite direct. Since α_i satisfy

$$\alpha_i(X)\phi_i(X) = f(X)\phi_i(X)$$

it follows that

$$\alpha_i(q) = f(q)$$

for q in $\text{supp}\phi_i$. Hence to determine consistency of $f = 0$ we can search over index i and the support of ϕ_i , independently for each i , such that $f(q) = 0$. We write this explicitly as a corollary to the above proposition.

Corollary 3. $f(X) = 0$ is consistent iff there exists an index i and an assignment q in $\text{supp}\phi_i$ such that $f(q) = 0$.

This result observes that $f = 0$ is consistent iff there is a zero of f in the support of at least one ϕ_i in the ON set Φ . This has important computational consequences depending on the choice of the ON set for computing solutions of a Boolean system (1).

4.2 Consistency of the Boolean system

To write the consistency condition for the Boolean system (1) we first express the individual functions f_i, g_i in ON expansion in terms of an ON set Φ . Let these expansions be

$$\begin{aligned} f_i(X) &= \sum_{j=1}^m \alpha_{ij}(X) \phi_j(X) \\ g_i(X) &= \sum_{j=1}^m \beta_{ij}(X) \phi_j(X) \end{aligned} \quad (20)$$

for $i = 1, \dots, N$.

For the case of systems following theorem shows that a search for solutions can be carried out without computing the single function F .

Theorem 2. Let a Boolean system (1) has ON expansions (20) of its functions. Then the system (1) is consistent iff there exists an index k in $[1, m]$ and an assignment q in $\text{supp}\phi_k$ such that

$$\alpha_{ik}(q) = \beta_{ik}(q) \forall i = 1, \dots, N$$

Every solution of a consistent system (1) arises this way.

Proof: Consider the single equation $F(X) = 0$ equivalent to the system (1) and the ON expansions of the individual functions (20) where

$$\begin{aligned} F(X) &= \sum_{i=1}^N (f_i(X) \oplus g_i(X)) \\ &= \sum_{j=1}^m \sum_{i=1}^N (\alpha_{ij}(X) \oplus \beta_{ij}(X)) \phi_j(X) \end{aligned}$$

Using the identities above for ON expansion w.r.t. Φ and the result of proposition 5 it follows that when $F(X) = 0$ is consistent and q is a solution, there is an index k in $[1, m]$ such that q belongs to $\text{supp}\phi_k$ and satisfies

$$\sum_{i=1}^N (\alpha_{ik}(q) \oplus \beta_{ik}(q)) = 0$$

which proves necessity of the condition and shows that every solution of the system satisfies this condition.

Conversely let there exists k in $[1, m]$ and a q in $\text{supp}\phi_k$ which satisfies

$$\alpha_{ik}(q) = \beta_{ik}(q) \forall i = 1, \dots, N$$

then from the (20) and noting that $\phi_i(q) = 0$ for $i \neq k$ it follows that

$$f_i(q) = g_i(q) \forall i = 1, \dots, N$$

Hence q is a solution of the system (1) which proves sufficiency and shows that every solution of a consistent system arises this way. \square

Remark 1. The proof above shows that once the ON set Φ is chosen the search space of solutions can be decomposed to subsets $\text{supp } \phi_i$ and further the resultant assignments satisfying the system can be found by evaluation and checking whether $f_i(q) = g_i(q)$. Hence the computation of co-efficient functions α_i, β_i is not required.

This theorem with the above remark forms a basis of a computational procedure developed in the next which is useful for writing parallel computational algorithms.

4.3 Computational procedure for Boolean systems

Solving Boolean systems by a scalable process is the central goal of practical computation. By scalable it is meant that the process works efficiently even when the size of the system is large enough to solve real industrial problems, as well as is able to utilize multiple parallel computations and works efficiently even over large number of such computing nodes (as are available in current technology). ON expansion based consistency condition developed above helps achieve scalability since the consistency of the original system and its solutions are determined independently from decomposed systems after substituting the assignments from supports of the ON functions. Before writing this process formally as an algorithm we need to take into account assignments arising from or solutions of some of the simplest Boolean systems which need not be expanded by ON sets to solve them and the system can be reduced after such trivial assignments. As a partial list of such systems and reductions consider

1. Number of variables $n = |X|$ as well as equations N in (1) is small. In such a case all solutions and consistency can be searched over B_0^n by brute force search.
2. Equations of the type $l_i = l_j$ in literals. If such an equation arises one variable in the system is reduced.
3. Unit clauses such as $l = 1$, trivial equations such as,

$$\sum l_i = 0, \prod l_i = 1$$

all are equations for which assignments are trivially determined.

We shall denote the function $\text{Trivsolve}()$ as a generic function which makes trivial assignments of variables whenever possible thereby reducing the system to a new system in which assigned variables are removed. This reduced system is considered an output of this function. The algorithm which carries out ON decomposition can now be written as follows.

Algorithm 1 (Decomposition). $\text{Decompose}()$

1. **Input** System denoted by (S, X) as in (1) with variables X . n_0 the largest number of variables below which the solution of S or its inconsistency can be determined by brute force search.
2. $n = |X|$, **while** $n > n_0$ **repeat**
3. Choose an ON set of $\Phi = \{\phi_i, i = 1, \dots, m\}$

4. For each i and q in $\text{supp } \phi_i$ determine the collection of all systems $S(q)$. Determine the variables $X(q)$ of $S(q)$.
5. Distribute each of these systems to an independent node for independent computation

$$(S, X) \leftarrow (S(q), X(q))$$

6. **return** system to be solved: (S, X) .

The main algorithm is now as follows

Algorithm 2 (ON decomposition based solver). BoolSolve()

1. Input (S, X) , n_0
2. $n = |X|$ **while** $n > n_0$ **repeat**
3. TrivSolve(S, X)
4. Decompose(S, X)
5. At each node **if** $n \leq n_0$ **return** solution of the system or set flag $SAT = F$, broadcast flag.

The ON expansion thus plays the role of decomposing the original system to a smaller size for independent parallel computation. In practice much of the efficiency can be gained by heuristics in deciding the ON set as well as in strategies for reducing variables during trivial solutions.

5 Decomposition and reduction of systems at partial assignments

In a general Boolean algebra B , a characterization of ON sets is obtained in [3, Theorem 4.2] which is reproduced below for convenience. If $\{y_1, y_2, \dots, y_m\}$ is an ON set in B then there are elements of the form u_1, \dots, u_{m-1} in B such that

$$\begin{aligned} y_1 &= u'_1 \\ y_j &= u_1 u_2 \dots u'_j \text{ for } 2 \leq j \leq m-1 \\ y_m &= u_1 u_2 \dots u_{m-1} \end{aligned}$$

From this characterization of ON sets it can be shown as in [3, theorem 4.2, corollary] that ON sets Φ of any order m from 2 to 2^n exist and are related to the partition of the set $\{0, 2^{n-1}\}$. Let symbols μ denote minterms in n variables which are of the form

$$\mu = x_1^{a_1} \dots x_n^{a_n}$$

where $a_i = 0, 1$ and the literals satisfy $x^1 = x$, $x^0 = x'$ there are 2^n minterms which can be indexed by the set $\{0, 1, \dots, 2^n - 1\}$. Consider

$$\cup_{j=1}^m M_j$$

to be a partition of the set $\{0, 1, \dots, 2^{n-1}\}$. Then the set $\Phi = \{\phi_i, i = 1, \dots, m\}$ defined by

$$\phi_i(X) = \sum_j \mu_{ij}, j \in M_i$$

where μ_{ij} are minterms indexed according the the partition above is ON. Conversely to any ON set Φ there is a unique partition as above of $\{0, 1, \dots, 2^{n-1}\}$ which defines the functions ϕ_i in Φ . To explain these representations of ON sets we consider illustrative examples.

Example 1. In three variables x, y, z consider ON sets

1. Of order 2: x, x' , represented in minterms as

$$\begin{aligned} x &= xyz + xy'z + xyz' + xy'z' \\ x' &= x'yz + x'y'z + x'yz' + x'y'z' \end{aligned}$$

2. Of order 3: $x, x'y, x'y'$, represented in minterms as

$$\begin{aligned} x &= xyz + xy'z + xyz' + xy'z' \\ x'y &= x'yz + x'y'z' \end{aligned}$$

3. Of order 4: $x, x'y, x'y'z, x'y'z'$,

4. As there are 8 minterms other ON sets can be constructed by summing the minterms.

Due to this characterization of ON sets as sums of minterms the solution sets of equations of the form

$$\phi_i(X) = 1$$

is trivial and can be described as follows.

Proposition 6. Let Φ be an ON set of order m in $B(X)$ and ϕ_i an element, the equation $\phi_i(X) = 1$ is consistent. A string q in B^n satisfies $\phi_i(q) = 1$ iff there exists a minterm $\mu_{ij} \in M_i$ such that $\mu_{ij}(q) = 1$. All solutions of $\phi_i(q) = 1$ arise this way.

Proof: Consider first proving the second and third statements. Note that the ON set being reduced has no zero functions among its elements. Since

$$\phi_i = \sum_{j \in M_i} \mu_{ij}$$

for minterms μ_{ij} , if there is q such that $\phi_i(q) = 1$, then this implies $\mu_{ij}(q) = 1$ for some j . If μ_{ij} is the product

$$\mu_{ij} = x_1^{a_1} x_2^{a_2} \dots x_n^{a_n}$$

then $\mu_{ij}(q) = 1$ has the only solution given by assignments $x_j = 1$ if $a_j = 1$ and $x_j = 0$ otherwise. This proves the last two statements. Hence $\phi_i(q) = 1$ is always consistent which proves first statement. \square

ON sets of functions can be more generally created by products and sums of functions in partitions of ON sets as follows.

Proposition 7. Let B be any Boolean algebra and $X = X_1 \cup X_2$ where subsets X_1, X_2 may not be disjoint.

1. If Φ is an ON set in $B(X)$ of order m and

$$\Phi = \cup_{i=1}^t F_i$$

is a disjoint partition of Φ . Then the set $\{\psi_i\}$ defined by

$$\psi_i = \sum_{\phi_i \in F_i} \phi_i$$

is an ON set of order t .

2. If Φ_i is an ON set in $B(X_i)$ of order m_i for $i = 1, 2$ then $\{\phi_{1k}\phi_{2l}\}$ for ϕ_{ik} in Φ_i is an ON set of order m_1m_2 .

5.1 ON terms and Partial assignments

In general if Φ is an ON set of functions and ϕ is an element, as shown in above proposition there are multiple solutions to $\phi(q) = 1$. However if the set Φ consists of ON terms $\{t_1, t_2, \dots, t_m\}$ then these multiple solutions are characterized by unique partial assignments of variables defining t_i . For instance if x', xy', xyz', xyz is an ON set then $xy' = 1$ has all solutions given by $(x, y, z) = (1, 0, 0), (1, 0, 1)$. Thus the unique assignments of x, y are enough to construct all solutions by assigning free variables (z in this case) freely. For a term

$$t(X) = \prod_i x_i^{r_i}$$

The partial assignment defined by $t(X) = 1$ is $x_i = 1$ when $r_i = 1$ and $x_i = 0$ when $r_i = 0$. The well known concept of *ratio* f/t of a function $f(X)$ and a term $t(X)$ is defined as the function [2],

$$(f/t)(X) = f(t(X) = 1)$$

where the partial assignments of variables in $t(X)$ are substituted in f while f/t is a function of the free variables which do not appear in the term t . As a useful notation for assignments of variables defined by partial assignments, consider a term $t(X)$ involving a subset of variables X_1 in X and X_2 denote rest of the (free) variables. Denote the partial assignments (that of X_1) which satisfy $t(X) = 1$ by $q(t)$. If a tuple q_2 are assignments of the variables X_2 , denote by $(q(t)||q_2)$ the assignment of X which is one of the solutions of $t(X) = 1$. The assignments of individual variables x_i at index i are suppressed in this notation as this an algorithmic task which will be incorporated in the computation.

Using partial assignments we can restate the consistency of systems (1) from theorem 2 as follows.

Corollary 4. Let $T = \{t_1, t_2, \dots, t_m\}$ be an ON set of terms. The system (1) is consistent iff there is an index k in $[1, m]$ such that the system

$$(f_i/t_k) = (g_i/t_k), i = 1, \dots, N$$

is consistent. If q is a solution of this system, then $(q(t_k)||q)$ gives a solution of the system (1). every solution of a consistent system (1) arises in this form.

Proof: The ON set is now T of terms. Hence by the theorem there exists an index k such that for q in $\text{supp } t_k(X)$ the equations are satisfied

$$f_i(q) = g_i(q) \forall i = 1, \dots, N$$

But q arises as $(q(t_k) \parallel q_2)$ where q_2 satisfies

$$(f_i/t_k)(q_2) = (g_i/t_k)(q_2) \forall i = 1, \dots, N$$

This proves that the $(f_i/t_k) = (g_i/t_k)$ is consistent with solution q_2 and every solution is of this form. \square

As a compact notation let the system (1) be denoted S and the component systems in ON expansion in terms of T be denoted S/t_k .

5.2 An example of computing rational solutions

We show an application of the ON term based procedure stemming from corollary 4 to computation of the roots of an algebraic equation over a finite field. Consider an elliptic curve expressed in Weierstrass form with co-efficients defined over the finite field $K = \mathbb{F}_{2^3}$ by

$$E \quad y^2 + xy + x^3 + (1 + \theta)x^2 + \theta = 0$$

where K is described in a polynomial basis defined by the irreducible polynomial $\theta^3 + \theta + 1$. Hence we have $\theta^3 = \theta + 1$. We want to compute all solutions of this equations (points (x, y) in K^2 constituting the elliptic curve E) in K . Note that it is not guaranteed that E is non empty.

A Boolean equation approach to solve this problem is to express variables x, y in K in co-ordinates $x = a + b\theta + c\theta^2$, $y = d + e\theta + f\theta^2$ in \mathbb{F}_2 treated as the Boolean algebra B_0 . This defines a Boolean system of equations in unknowns a, \dots, f in B_0 . The solution set of this system thus gives all points on E and when the equations are inconsistent E is empty. Consider the ON set of terms

$$T = \{c, c'b, c'b'a, c'b'a'\}$$

When partial assignments defined by these terms are substituted in the equation of E we get independent quadratic equations over K such as

$$pT^2 + qT + r = 0$$

with $p \neq 0$. This equation has well known solutions. For $q = 0$ the multiple solution $\sqrt{r/p}$. For $q \neq 0$, defining $s = pr/q^2$ solution exists iff $\text{Tr } s = 0$. The two solutions are $u, u + 1$ where u is a nonzero solution in the kernel of the Artin-Schreier map $x \rightarrow x^2 + x$. Computation of this kernel can be yet another Boolean system problem in the chosen basis for K . In this example we shall only show how different substitutions of partial assignments result into checking existence via trace evaluation as above.

Let the ON terms in T be indexed as $t_i, i = 1, \dots, 4$. We denote substitutions by E/t_i

1. E/t_4 . In this $x = 0$. The solutions exists, $y = \sqrt{\theta}$.
2. E/t_3 . In this $x = 1$. The equation is $y^2 + y + (1 + \theta + \theta^2)$. $\text{Tr } (1 + \theta + \theta^2) = 1$, hence no solution.

3. E/t_2 . In this $x = a + \theta$. Hence two cases are studied $x = \theta$ and $x = 1 + \theta$. In both cases the s calculated has trace zero. Hence solutions exist for both $a = 0, 1$.
4. E/t_1 . Here $x = a + b\theta + \theta^2$. Hence we can further expand the equation $E/t_1 = 0$ w.r.t. ON terms a', ab', ab . Let these be denoted by indexing $E/t_{11}, E/t_{12}, E/t_{13}$ respectively with values of x as in the last column of the following table

$$\begin{array}{rcl}
E/t_{11} & b = 1 & (\theta + \theta^2) \\
& b = 0 & \theta^2 \\
E/t_{12} & & (1 + \theta^2) \\
E/t_{13} & & (1 + \theta + \theta^2)
\end{array}$$

Evaluation of traces of s defined by x in each of the substitutions of x above gives all remaining solutions.

5.3 Algorithm for CNF-SAT

Finally we shall discuss the ON expansion based algorithm for a CNF satisfiability problem. Let C or $C(X)$ denote the set of all clauses over the set of literals denoted X and $\mathcal{S}(C)$ denote the set of all assignments for satisfying all clauses in C . If x in X is a pure literal (i.e. no clause in C contains x') then $\mathcal{S}(C)$ is non empty iff for the set C/x which equals $\{C_j(x = 1) | C_j \in C\}$ the set $\mathcal{S}(C/x)$ is non empty. Let $t_{r+1} = x_1 \dots x_r$ be the term denoting product of all pure literals in C and consider the ON set

$$T = \{t_1, \dots, t_{r+1}\}$$

where

$$\begin{array}{rcl}
t_1 & = & x'_1 \\
t_2 & = & x_1 x'_2 \\
\vdots & = & \vdots \\
t_r & = & x_1 x_2 \dots x'_r \\
t_{r+1} & = & x_1 x_2 \dots x_r
\end{array} \tag{21}$$

Then the following proposition follows.

Proposition 8. With the r positive variables and ON terms defined above C is satisfiable iff C/t_{r+1} is satisfiable. All satisfying assignments of C when it is satisfiable are obtained as the union of all assignments $\mathcal{S}(C/t_i)$ along with the partial assignments $q(t_i)$.

We shall illustrate this process by an example.

5.3.1 Example of CNF-SAT 1

Consider the CNF set C given by the matrix whose first row gives column indices indexing the variables. Subsequent rows indicate clauses. In this notation 1 below a column i indicates x_i

and -1 indicates x'_i term in the clause.

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & 0 & -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & -1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & -1 & 0 & -1 & -1 & 0 & 1 \\ 0 & -1 & 0 & 1 & 0 & 0 & -1 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Thus x_1, x'_3 are pure. We choose ON terms

$$\begin{aligned} t_1 &= x'_1 \\ t_2 &= x_1 x_3 \\ t_3 &= x_1 x'_3 \end{aligned}$$

After assigning pure literals $x_1 = x'_3 = 1$, the CNF set C/t_3 is given by

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 0 & -1 & 0 & 1 & 0 & 0 & -1 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

which is clearly satisfiable by $(1, 0, 0, 0, d, d, d, d)$. Hence C is satisfiable.

Thus in CNF satisfiability if we can assume that all pure literals have been assigned already or that there are no pure literals to be assigned. Similarly assume that all unit clauses $\{l\}$ have been eliminated by assignments, then the above corollary can be written analogously for the CNF set with no pure literals and unit clauses as follows.

Corollary 5. Let C be a CNF set with no pure literals or unit clauses and let $T = \{t_1, t_2, \dots, t_m\}$ be an ON set of terms. Then the set C is satisfiable iff there is an index k in $[1, m]$ such that the system

$$C/t_k$$

is satisfiable. If q is a satisfying assignment of this set, then $(q(t_k)||q)$ gives an assignment satisfying C . Every solution of a satisfiable CNF set C arises in this form.

5.4 ON expansion based algorithm for CNF-SAT

Combining the above two cases we get a procedure for decomposition and solving satisfiability of a CNF set which is a generalization of the DPLL algorithm. This generalization is in respect of the splitting rule of DPLL which is defined by the ON functions x, x' relative to a single variable x , while the decomposition considered in the following algorithm extends the notion of splitting relative to an ON set of terms in many variables.

Let (C, X) denote the CNF set C with the list of literals X . The algorithm described next does the task of decomposition of the SAT problem data (C, X) to sub-problems by choosing an ON set of terms and distributes the sub-problems to independent nodes for computation of solutions. The parameter n_0 is the maximum number of variables X , below which the SAT problem is solved by direct search over all assignments.

Algorithm 3 (Decomposition and Distribution). $\text{Decompose}((C, X))$

1. **Input** $(C, X), n_0$
2. **while** $|X| = n > n_0$, **repeat**
3. Choose an ON set of terms T over a subset of *literals* in X .
4. *Decompose*: compute sub-problems

$$C_i = C/t_i \text{ for } t_i \in T$$

compute resultant literals X_i after partial assignment such that $t_i = 1$.

5. *distribute* (C_i, X_i) to independent nodes for independent computation.

The above algorithm for distribution is then used in the main algorithm

Algorithm 4 (Main). SolveSAT((C, X))

1. **Input** $(C, X), n_0$
2. **while** $|X| > n_0$, **repeat**
3. Determine all *unit clauses* with subset of literals X_1 . Assign $x = 1$ for all $x \in X_1$.

$$(C, X) \leftarrow (C(x = 1, x \in X_1), X - X_1)$$

4. Determine *pure literals* x_1, \dots, x_r , make partial assignments $x_i = 1$ for $i = 1, \dots, r$ denoted by $t_{r+1} = 1$ as in (21) and denote \tilde{X} the set of un-assigned literals

$$(C, X) \leftarrow (C/t_{r+1}, \tilde{X})$$

5. *Decompose*((C, X))
6. At each independent node: **if** $|X| \leq n_0$ solve the satisfiability problem. **return** solution at the node **else** assign flag $SAT = F$ broadcast flag.

We illustrate the algorithm with an example.

5.4.1 Example for CNF-SAT 2

Consider the CNF set given by the matrix C , first row denoting column indices,

$$C = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 1 & -1 & 0 & 0 & 1 & 0 & 0 & -1 \\ -1 & 0 & 1 & -1 & 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & -1 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & -1 & 0 \end{bmatrix}$$

and variables $X = \{x_1, \dots, x_8\}$ indexed by columns of C . There are no positive or pure literals. We choose ON set

$$\{x_1, x'_1 x_2, x'_1 x'_2\}$$

and decompose into

$$C_1 = C/x_1 = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 0 & 0 & 1 & -1 & 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & -1 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & -1 & 0 \end{bmatrix}$$

$$C_2 = C/x'_1x_2 = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & -1 & -1 & 0 \end{bmatrix}$$

$$C_3 = C/x'_1x'_2 = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & -1 & 0 \end{bmatrix}$$

In C_1 pure literals are assigned as $x_2 = x_8 = 1$. This reduces C_1 to

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 0 & 0 & 1 & -1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & -1 & 0 \end{bmatrix}$$

This leads to pure literal assignments $x_3 = x_5 = 1$. Hence other variables x_4, x_6, x_7 can be assigned arbitrary. Hence (C, X) is SAT even without checking the other cases C_2, C_3 .

6 Conclusions

This paper resolves the problem of solving Boolean systems of equations in many variables and B_0 co-efficients using general ON expansion without converting the system to single equation. The method also characterizes all solutions of such systems. As a central idea the ON expansion splits the original problem into a smaller sub-problems indexed by ON functions whose solutions need be searched only on the support of the specific ON function. Since the assignments in the support of an ON function are known apriori (or are easy to determine on the fly) the search of solution of the smaller problem is simplified. Further, since the solutions of the smaller problems are computed independently, this algorithm is inherently parallel. Algorithms for solving Boolean systems can be improved in their performance further by incorporating conjugate problems during computation. This aspect needs to be explored further.

The special case of expansion when the ON functions are ON terms is also shown to be useful in deciding satisfiability and characterization of solutions. This special procedure also leads to a generalization of the DPLL algorithm where the splitting stage is extended over many variables. The CNF-SAT algorithm developed using this special procedure is most valuable for decomposition of the problem for parallel computation. Methods developed in this paper, it is hoped, shall be useful for devising scalable parallel approaches of Boolean satisfiability problems by incorporating heuristics in constructing ON sets of functions.

Acknowledgements

Supported by the project grant 11SG010 of IRCC of IIT Bombay. Author gratefully acknowledges enlightening comments by Professor Rudeanu leading to improvements in the paper.

References

- [1] George Boole. An Investigation of the Laws of thought. Walton, London, 1854.
- [2] F. M. Brown. Boolean reasoning. The logic of Boolean equations. Dover, 2006.
- [3] Sergiu Rudeanu. Boolean functions and equations. North Holland, Amsterdam, 1974.
- [4] Sergiu Rudeanu. Lattice functions and equations. Springer Verlag, London, 2001.
- [5] A. Biere, M. Heule, Hans van Maaren, T. Walsh (Eds). Handbook of Satisfiability. IOS Press, 2009.
- [6] Yves Crama and Peter Hammer. Boolean functions. Theory, algorithms and applications. Encyclopedia of Mathematics and its applications, vol.142. Cambridge, 2011.
- [7] Gregory Bard. Algebraic cryptanalysis. Springer 2009.
- [8] Marc Mezard and Andrea Montanari. Information, Physics and Computation. Oxford University Press, 2009.
- [9] Youssef Hammadi and C. M. Wintersteiger. Seven challenges in parallel SAT solving. Challenge paper AAAI 2012 Sub-Area spotlights track. Association of Advancement of Artificial Intelligence.
- [10] Kohavi and Jha, Switching and automata theory, Cambridge 2008.
- [11] Generalization of Boole-Shannon expansion, consistency of Boolean equations and elimination by orthonormal expansion, arXiv.org:1306.2484v3,[cs.CC], December 6, 2013.